University of Mary Washington
Eagle Scholar

Student Research Submissions

Spring 4-25-2017

Algorithms to Approximate Solutions of Poisson's Equation in Two and Three Dimensions

Rachelle Serena Dambrose

Follow this and additional works at: https://scholar.umw.edu/student_research

 Part of the Mathematics Commons

Recommended Citation

Dambrose, Rachelle Serena, "Algorithms to Approximate Solutions of Poisson's Equation in Two and Three Dimensions" (2017). *Student Research Submissions*. 161.
https://scholar.umw.edu/student_research/161

This Honors Project is brought to you for free and open access by Eagle Scholar. It has been accepted for inclusion in Student Research Submissions by an authorized administrator of Eagle Scholar. For more information, please contact archives@umw.edu.

ALGORITHMS TO APPROXIMATE SOLUTIONS OF POISSON'S EQUATION IN TWO AND THREE DIMENSIONS

An honors paper submitted to the Department of Mathematics
of the University of Mary Washington
in partial fulfillment of the requirements for Departmental Honors

Rachelle Serena Dambrose
April 2017

By signing your name below, you affirm that this work is the complete and final version of your paper submitted in partial fulfillment of a degree from the University of Mary Washington. You affirm the University of Mary Washington honor pledge: "I hereby declare upon my word of honor that I have neither given nor received unauthorized help on this work."

Rachelle Dambrose
(digital signature)

04/25/17

Algorithms to Approximate Solutions of Poisson's Equation in Two and Three Dimensions

Rachelle Dambrose

submitted in partial fulfillment of the requirements for Honors in Mathematics at the University of Mary Washington

Fredericksburg, Virginia

April 2017

This thesis by **Rachelle Dambrose** is accepted in its present form as satisfying the thesis requirement for Honors in Mathematics.

Date	Approved
_____	_____
	Jangwoon Lee, Ph.D. thesis advisor
_____	_____
	Debra L. Hydorn, Ph.D. committee member
_____	_____
	J. Larry Lehman, Ph.D. committee member

Contents

Abstract

The focus of this research was to develop numerical algorithms to approximate solutions to Poisson's equation in two and three dimensions. Numerical analysis of partial differential equations is vital to understanding and modeling these complex problems. A finite difference approximation of Poisson's equation can be used to form a system of linear equations of solutions through a region. A computer program was developed to solve this system with inputs such as boundary conditions and a nonhomogenous source function. Approximate solutions were compared with exact solutions to prove their accuracy. The program was tested with an increasing number of subintervals to ensure that the approximations got closer to the actual solution. Then, an experiment was performed to find the temperatures through a heated piece of aluminum foil to show how this approximation can predict real world phenomenon.

1 Introduction

Poisson's Equation is a steady-state, time-independent variation of Laplace's equation [2].

$$\nabla^2 u = Q \tag{1}$$

The goal is to solve for u in a given region, R. Here, Q is a nonhomogenous source function through the region. Boundary conditions, α, of our region must also be given. Variations of Poisson's Equation can be used to model temperature, concentration of particles after diffusion, electrostatic potential, and Newtonian gravity potential [3].

Although actual solutions to Poisson's equations are known, these solutions are complicated and difficult to calculate, because they contain multiple double infinite sums. Numerical approximations become necessary to efficiently find solutions to mathematical models. The goal of this project was to create algorithms that can be used to approximate solutions to Poisson's equation in two and three dimensions. In two dimensions, we approximate solutions in rectangular domains. In three dimensions, we approximate solutions in rectangular prism domains. To create this program, we find and solve finite difference approximations of Poisson's equation to identify patterns that will eventually act as algorithms in the final program.

2 Two Dimension Case

2.1 Finite Difference Approximation in 2D

Consider a function on a two dimensional plane of length L and height H given by

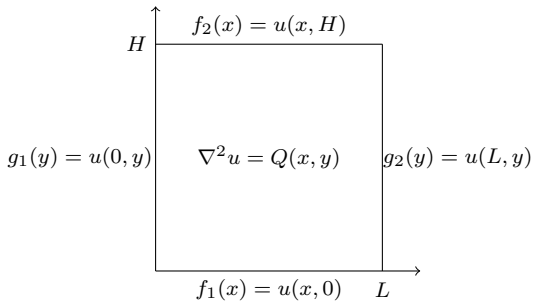
$$\begin{aligned} \nabla^2 u &= Q(x,y) \\ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} &= Q(x,y) \end{aligned} \tag{2}$$

Here, $Q(x,y)$ is the surface function of the rectangular domain, where $0 \leq x \leq L$ and $0 \leq y \leq H$. Additionally, we must know the boundary conditions, α, of our problem, where $x = 0$, $x = L$,

$y = 0$, and $y = H$. These can be given as four functions:

$$u(x,0) = f_1(x) \qquad u(x,H) = f_2(x)$$
$$u(0,y) = g_1(y) \qquad u(L,y) = g_2(y)$$

Our surface can be sketched as:



We can approximate the second derivatives, $\nabla^2 u$, at any point in our region, (x_i, y_l), with a finite difference approximation [2].

$$\nabla^2 u \approx \frac{u(x_i + \Delta x, y_l) + u(x_i - \Delta x, y_l) - 2u(x_i, y_l)}{(\Delta x)^2} + \frac{u(x_i, y_l + \Delta y) + u(x_i, y_l - \Delta y) - 2u(x_i, y_l)}{(\Delta y)^2} \tag{3}$$

If we take $\Delta x = \Delta y$, then we can simplify equation (3) as

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial^2 y} \approx \frac{u(x_i + \Delta x, y_l) + u(x_i - \Delta x, y_l) + u(x_i, y_l + \Delta y) + u(x_i, y_l - \Delta y) - 4u(x_i, y_l)}{(\Delta x)^2} \tag{4}$$

Using equation (4), Poisson's Equation in two dimensions, (2), can be approximated as

$$\frac{u(x_i + \Delta x, y_l) + u(x_i - \Delta x, y_l) + u(x_i, y_l + \Delta y) + u(x_i, y_l - \Delta y) - 4u(x_i, y_l)}{(\Delta x)^2} = Q(x,y) \tag{5}$$

Now, consider a new notation for equation (5). Let $x_i + \Delta x$ be denoted as x_{i+1} and $y_l + \Delta y = y_{l+1}$. Any solution $u(x_i, y_l)$ can be denoted as u_{il}. We can rewrite equation (5) with this new notation.

$$(\Delta x)^2 \cdot Q(x_i, y_l) = u_{i+1,l} + u_{i-1,l} + u_{i,l+1} + u_{i,l-1} - 4u_{i,l} \tag{6}$$

2.2 Square Domains

Consider a square domain, where $H = L$. The surface is split into $n = \frac{L}{\Delta x} = \frac{H}{\Delta y}$ subintervals on the x-axis and y-axis. To identify the pattern in solving approximations for any n, consider how we will approximate $u(x,y)$ with $n = 5$ subintervals.

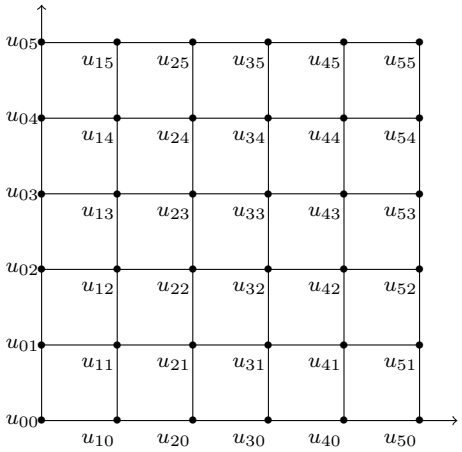


Figure 1: Plane in 5 Subintervals

Note that there are points of our square that are already known. Given the boundary conditions, we know solutions for points where $x = 0$, $x = L$, $y = 0$, and $y = H$. We only need to solve for solutions at the 16 internal points of the square. For convenience, we will label these solutions as v_r. All we know at these internal points is the source function given by $Q(x, y)$. We can rewrite Figure 1 as:

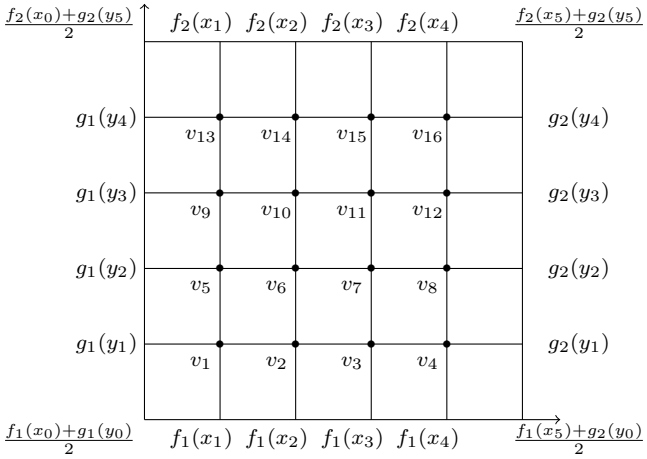


Figure 2: Plane with Boundary Conditions

Note that we cannot assume that the boundary conditions at the corners of our plane are equivalent. It is possible that $f_1(x_0) \neq g_1(y_0)$, $f_1(x_n) \neq g_2(y_0)$, $f_2(x_0) \neq g_2(y_n)$, and $f_2(x_n) \neq g_2(y_n)$. We will take the solution at each corner as the average of the two overlapping boundary conditions, so solutions over the square or rectangular domain are continuous.

We can solve for solutions u_{il}, where $l = 1, 2, \cdots, (n-1)$ and $i = 1, 2, \cdots, (n-1)$ by using equation

(6) to create a system of linear equations.

$$
\begin{aligned}
&\underline{l=1}\\
&\quad i=1 \qquad u_{2,1}+u_{0,1}+u_{1,2}+u_{1,0}-4u_{1,1}=(\Delta x)^2\cdot Q(x_1,y_1)\\
&\quad i=2 \qquad u_{3,1}+u_{1,1}+u_{2,2}+u_{2,0}-4u_{2,1}=(\Delta x)^2\cdot Q(x_2,y_1)\\
&\quad i=3 \qquad u_{4,1}+u_{2,1}+u_{3,2}+u_{3,0}-4u_{3,1}=(\Delta x)^2\cdot Q(x_3,y_1)\\
&\quad i=4 \qquad u_{5,1}+u_{3,1}+u_{4,2}+u_{4,0}-4u_{4,1}=(\Delta x)^2\cdot Q(x_4,y_1)\\
&\underline{l=2}\\
&\quad i=1 \qquad u_{2,2}+u_{0,2}+u_{1,3}+u_{1,1}-4u_{1,2}=(\Delta x)^2\cdot Q(x_1,y_2)\\
&\quad i=2 \qquad u_{3,2}+u_{1,2}+u_{2,3}+u_{2,1}-4u_{2,2}=(\Delta x)^2\cdot Q(x_2,y_2)\\
&\quad i=3 \qquad u_{4,2}+u_{2,2}+u_{3,3}+u_{3,1}-4u_{3,2}=(\Delta x)^2\cdot Q(x_3,y_2)\\
&\quad i=4 \qquad u_{5,2}+u_{3,2}+u_{4,3}+u_{4,1}-4u_{4,2}=(\Delta x)^2\cdot Q(x_4,y_2)\\
&\underline{l=3}\\
&\quad i=1 \qquad u_{2,3}+u_{0,3}+u_{1,4}+u_{1,2}-4u_{1,3}=(\Delta x)^2\cdot Q(x_1,y_3)\\
&\quad i=2 \qquad u_{3,3}+u_{1,3}+u_{2,4}+u_{2,2}-4u_{2,3}=(\Delta x)^2\cdot Q(x_2,y_3)\\
&\quad i=3 \qquad u_{4,3}+u_{2,3}+u_{3,4}+u_{3,2}-4u_{3,3}=(\Delta x)^2\cdot Q(x_3,y_3)\\
&\quad i=4 \qquad u_{5,3}+u_{3,3}+u_{4,4}+u_{4,2}-4u_{4,3}=(\Delta x)^2\cdot Q(x_4,y_3)\\
&\underline{l=4}\\
&\quad i=1 \qquad u_{2,4}+u_{0,4}+u_{1,5}+u_{1,3}-4u_{1,4}=(\Delta x)^2\cdot Q(x_1,y_4)\\
&\quad i=2 \qquad u_{3,4}+u_{1,4}+u_{2,5}+u_{2,3}-4u_{2,4}=(\Delta x)^2\cdot Q(x_2,y_4)\\
&\quad i=3 \qquad u_{4,4}+u_{2,4}+u_{3,5}+u_{3,3}-4u_{3,4}=(\Delta x)^2\cdot Q(x_3,y_4)\\
&\quad i=4 \qquad u_{5,4}+u_{3,4}+u_{4,5}+u_{4,3}-4u_{4,4}=(\Delta x)^2\cdot Q(x_4,y_4)
\end{aligned}
\tag{7}
$$

We can now assign the boundary conditions to our linear equations. For example, we already know that $u_{1,0}=f_1(x_1)$. We must also reassign solutions $u_{i,l}$ to corresponding $v_1,\ v_2,\cdots,\ v_{(n-1)^2}$. Consider how system (7) changes where $l=1$. (See Figure 1 and 2.)

$$
\begin{aligned}
&\underline{l=1}\\
&\quad i=1 \quad v_2\ +g_1(y_1)+v_5+f_1(x_1)-4v_1=(\Delta x)^2\cdot Q(x_1,y_1)\\
&\quad i=2 \quad v_3\ +\ v_1\ +\ v_6\ +f_1(x_2)-4v_2=(\Delta x)^2\cdot Q(x_2,y_1)\\
&\quad i=3 \quad v_4\ +\ v_2\ +\ v_7\ +f_1(x_3)-4v_3=(\Delta x)^2\cdot Q(x_3,y_1)\\
&\quad i=4 \quad g_2(y_1)+v_3+v_8+f_1(x_4)-4v_4=(\Delta x)^2\cdot Q(x_4,y_1)
\end{aligned}
\tag{8}
$$

We can find our solutions by converting this system (7) to a single equation of matrices and vectors.

$$A\vec{v}+\vec{b}=(\Delta x)^2\vec{q} \tag{9}$$

The goal is to solve for $\vec{v}$, the vector of approximate solutions inside the plane. A is the coefficient matrix of these solutions, $\vec{b}$ is the vector of boundary conditions, and $\vec{q}$ contains the source function, Q, at each point. We can solve for solutions by using Gaussian elimination on equation (9). In order to write a program that solves this problem, we must first identify the patterns of A, $\vec{b}$, and, $\vec{q}$.

2.2.1 A Matrix Pattern

In order to find A for any number of subintervals, consider A where $n = 5$, the case started above. The A matrix would be:

$$A = \left[\begin{array}{cccc|cccc|cccc|cccc} -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 \end{array}\right]$$

Note that A is a block matrix, meaning it is a matrix comprised of smaller matrices. In this case, A is a $4 \times 4 = (n-1) \times (n-1)$ matrix of $4 \times 4 = (n-1) \times (n-1)$ submatrices. On the main diagonal, we see a unique submatrix, A'.

$$A' = \begin{bmatrix} -4 & 1 & 0 & 0 \\ 1 & -4 & 1 & 0 \\ 0 & 1 & -4 & 1 \\ 0 & 0 & 1 & -4 \end{bmatrix}$$

On either side of the main diagonal, there is the 4×4 identiy matrix I. The rest of the matrices are the 4×4 zero matrix. So, as a block matrix, A can be simplified as:

$$A = \begin{bmatrix} A' & I & 0 & 0 \\ I & A' & I & 0 \\ 0 & I & A' & I \\ 0 & 0 & I & A' \end{bmatrix}$$

Now let us consider how to apply this pattern to A for a general number of subintervals n.

1. For any square region separated into n subintervals, A is a $(n-1) \times (n-1)$ matrix of $(n-1) \times (n-1)$ submatrices.

2. Within A, there is a notable $(n-1)\times(n-1)$ matrix A' with -4 on the main diagonal, 1 on either side of the main diagonal, and 0 everywhere else.

$$A' = \begin{bmatrix} -4 & 1 & 0 & \dots & 0 \\ 1 & -4 & 1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 1 & -4 & 1 \\ 0 & \dots & 0 & 1 & -4 \end{bmatrix}$$

3. In A, A' lies on the main diagonal. On either side of the main diagonal, we see the $(n-1)\times(n-1)$ identity matrix, I. All other submatrices are the $(n-1)\times(n-1)$ zero matrix, 0.

$$A = \begin{bmatrix} A' & I & 0 & \cdots & 0 \\ I & A' & I & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I & A' & I \\ 0 & \cdots & 0 & I & A' \end{bmatrix}$$

2.2.2 Boundary Condition and Source Vector Pattern

Let us consider how x_i and y_l change as we iterate through the vectors $\vec{b}$ and $\vec{q}$. We can find i and l at any solution $v_r = u_{il} = u(x_i, y_l)$ as

$$i = (r-1) \pmod{(n-1)} + 1 \qquad l = \left\lfloor \frac{r-1}{n-1} \right\rfloor + 1 \tag{10}$$

Here, $\left\lfloor \frac{r-1}{n-1} \right\rfloor$ is the floor of $\frac{r-1}{n-1}$, or the largest integer less than or equal to $\frac{r-1}{n-1}$. Note how solutions of internal points are labeled in figures 1 and 2. Solutions at the first few points are denoted as $v_1 = u_{1,1},\ v_2 = u_{2,1},\ \dots,\ v_{n-1} = u_{n-1,1}$. Then, at v_n, l increments to 2 and i returns to 1, so that $v_n = u_{1,2},\ v_{n+1} = u_{2,2},\ \dots$. Thus, i iterates as 1, 2, 3, ... , $n-1$, 1, 2, ... , returning to 1 for any v_r where $r-1 \equiv 0 \pmod{(n-1)}$. Alternately, l increments as 1, 1, ..., 1, $n-1$ times and increments to 2 for any v_r such that $r-1 \equiv 0 \pmod{(n-1)}$.

Recall from systems (7) and (8) how we can form the source function vector, $\vec{q}$, and boundary

condition vector, $\vec{b}$, where $n = 5$.

$$\vec{q} = \begin{pmatrix} Q(x_1, y_1) \\ Q(x_2, y_1) \\ Q(x_3, y_1) \\ Q(x_4, y_1) \\ Q(x_1, y_2) \\ Q(x_2, y_2) \\ Q(x_3, y_2) \\ Q(x_4, y_2) \\ Q(x_1, y_3) \\ Q(x_2, y_3) \\ Q(x_3, y_3) \\ Q(x_4, y_3) \\ Q(x_1, y_4) \\ Q(x_2, y_4) \\ Q(x_3, y_4) \\ Q(x_4, y_4) \end{pmatrix} \qquad \vec{b} = \begin{pmatrix} f_1(x_1) + g_1(y_1) \\ f_1(x_2) \\ f_1(x_3) \\ f_1(x_4) + g_2(y_1) \\ \hline g_1(y_2) \\ 0 \\ 0 \\ g_2(y_2) \\ \hline g_1(y_3) \\ 0 \\ 0 \\ g_2(y_3) \\ \hline f_2(x_1) + g_1(y_4) \\ f_2(x_2) \\ f_2(x_3) \\ f_2(x_4) + g_2(y_4) \end{pmatrix}$$

Consider how we can use the incrementing equations (10) to build $\vec{q}$ and $\vec{b}$. For example, consider the fourth entry of each vector, which corresponds to v_4. From equation (10), $i = (4-1) \pmod 4 + 1 = 4$ and $l = \left\lfloor \frac{4-1}{4} \right\rfloor + 1 = 0 + 1 = 1$. We find $v_4 = u_{41}$, the solution located at $(x_4,\ y_1)$. Thus, $\vec{q}$ holds the source function $Q(x_4,\ y_1)$. The point $(x_4,\ y_1)$ falls directly beside the $x = L$ and $y = 0$ boundaries and depends on two boundary conditions, $f_1(x)$ and $g_2(y)$. At $(x_4,\ y_1)$, we know the boundary conditions around this point are $f_1(x_4)$ and $g_2(y_1)$.

Now, consider how the boundary conditions appear in $\vec{b}$. Here, it is easiest to see the pattern when $\vec{b}$ is separated into 4 subvectors of length 4. $f_1(x)$ appears in the first 4 points, or the first subvector, whereas $f_2(x)$ appears in the last 4 points, or in the last subvector. $g_1(y)$ appears for v_1, v_4, v_9, and v_{13}, or the first entry for each subvector. Similarly, $g_2(y)$ appears for v_4, v_8, v_{12}, and v_{16}, or the last entry for each subvector.

Now, consider an algorithm for finding $\vec{b}$ using any number of subintervals n.

1. In a square domain, there are $(n-1)^2$ internal points, so $\vec{b}$ has length $(n-1)^2$. We will split this vector into $n-1$ subvectors of length $n-1$.

2. The first subvector holds all instances of the $f_1(x)$ boundary condition. This occurs for all v_r such that $v_r = u_{i1}$, or when $\left\lfloor \frac{r-1}{n-1} \right\rfloor + 1 = 1$. This makes sense, because only the first $n-1$ points lie against the $y = 0$ boundary. All instances of $f_2(x)$ occur in the last subvector or the last $n-1$ points. This occurs when $v_r = u_{i,(n-1)}$ or when $\left\lfloor \frac{r-1}{n-1} \right\rfloor + 1 = n-1$. This includes

all points on the top row of our domain, next to where $y = H$.

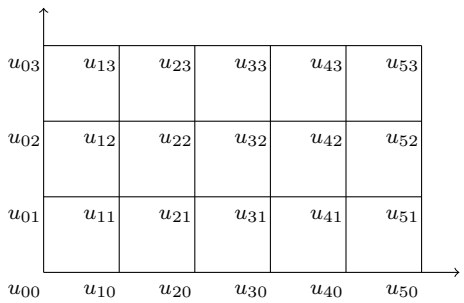
$$\begin{pmatrix} f_1(x_1) \\ f_1(x_2) \\ \vdots \\ f_1(x_{n-1}) \\ \hline \vdots \\ \hline f_2(x_1) \\ f_2(x_2) \\ \vdots \\ f_2(x_{n-1}) \end{pmatrix}$$

3. The $g_1(y)$ function appears in the first entry of each subvector. This occurs for every $v_r = u_{1l}$ or when $(r-1) \pmod{(n-1)} + 1 = 1$. Here, v_r are solutions for points that lie directly to the right of the $x = 0$ boundary. Similarly, $g_2(y)$ appears in the last entry of each subvector. This occurs for every $v_r = u_{(n-1),l}$ or when $(r-1) \pmod{(n-1)} + 1 = n-1$. These solutions lie directly to the left of the $x = L$ boundary.

$$\vec{b} = \begin{pmatrix} f_1(x_1) + g_1(y_1) \\ f_1(x_2) \\ \vdots \\ f_1(x_{n-2}) \\ f_1(x_{n-1}) + g_2(y_1) \\ \hline \vdots \\ \hline g_1(y_l) \\ 0 \\ \vdots \\ 0 \\ g_2(y_l) \\ \hline \vdots \\ \hline f_2(x_1) + g_1(y_{n-1}) \\ f_2(x_2) \\ \vdots \\ f_2(x_{n-2}) \\ f_2(x_{n-1}) + g_2(y_{n-1}) \end{pmatrix}$$

2.3 Rectangular Domains

Consider a more general case, where $H \neq L$. In order to keep $\Delta x = \Delta y$, we will have a different number of subintervals on the x and y axes, denoted below as n_x and n_y. The number of subintervals will be given as $n_x = \frac{L}{\Delta x}$ and $n_y = \frac{H}{\Delta y}$. Note that n_x and n_y are corresponding to n. Consider the case where $L = 5$, $H = 3$, and $\Delta x = \Delta y = 1$. Our rectangle is separated into $n_x = 5$ subintervals along the x axis, whereas the y-axis is separated into $n_y = 3$ subintervals.



This will change the pattern of A, $\vec{b}$, and $\vec{q}$. For this example, A is given as

$$A = \left[\begin{array}{cccc|cccc} -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -4 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 \end{array}\right]$$

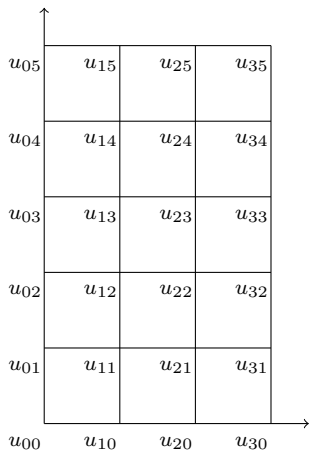
Note the pattern for this block matrix. A is a $2 \times 2 = (n_y - 1) \times (n_y - 1)$ matrix with $4 \times 4 = (n_x - 1) \times (n_x - 1)$ matrices as entries. Similar to the square case, there is a notable 4×4 matrix A' with -4 on its main diagonal and 1 on either side. In A, A' falls on the main diagonal with the 4×4 idenity matrix on either side.

The boundary condition vector for this rectangular domain example is

$$\vec{b} = \left(\begin{array}{c} f_1(x_1) + g_1(y_1) \\ f_1(x_2) \\ f_1(x_3) \\ f_1(x_4) + g_2(y_1) \\ \hline f_2(x_1) + g_1(y_2) \\ f_2(x_2) \\ f_2(x_3) \\ f_2(x_4) + g_2(y_2) \end{array}\right)$$

In this case, we can split $\vec{b}$ into $(n_y - 1) = 2$ subvectors of length $(n_x - 1) = 4$. Similar to the square case, $f_1(x)$ appears in the first subectvor and $f_2(x)$ appears in the last subvector. $g_1(y)$ appears in the first entry of every subvector, whereas $g_2(y)$ appears in the last entry of each subvector.

Alternately, consider a similar case, where $L = 3$, $H = 5$, $\Delta x = \Delta y = 1$, so $n_x = 3$ and $n_y = 5$. The plane is sketched below



For this similar rectangle, we find the A matrix as

$$A = \left[\begin{array}{cc|cc|cc|cc} -4 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & -4 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & -4 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & -4 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & -4 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & -4 \end{array}\right]$$

Here, A is a $4 \times 4 = (n_y - 1) \times (n_y - 1)$ block matrix of $2 \times 2 = (n_x - 1) \times (n_x - 1)$ submatrices. Again, we see the 2×2 matrix A' on the main diagonal, and the 2×2 identity matrix to either side.

The boundary condition vector for this rectangle is

$$\vec{b} = \left(\begin{array}{c} f_1(x_1) + g_1(y_1) \\ f_1(x_2) + g_2(y_1) \\ \hline g_1(y_2) \\ g_2(y_2) \\ \hline g_1(y_3) \\ g_2(y_3) \\ \hline f_2(x_1) + g_1(y_4) \\ f_2(x_2) + g_2(y_4) \end{array}\right)$$

Now, $\vec{b}$ can be broken into $4 = (n_y - 1)$ subvectors of length $2 = (n_x - 1)$.

The main difference between these rectangles is n_x and n_y, which drastically alters the shape of the A matrix and boundary condition vectors. Thus, we must redefine the pattern for creating the A matrix and $\vec{b}$ vector where $n_x \neq n_y$.

2.3.1 A Matrix Pattern

For any rectangular domain, we can construct A as:

1. Given H, L, and $\Delta x = \Delta y$, we find $n_x = \frac{L}{\Delta x}$ and $n_y = \frac{H}{\Delta y}$. A is a $(n_y - 1) \times (n_y - 1)$ block matrix made up of submatrices, size $(n_x - 1) \times (n_x - 1)$.

2. Define a $(n_x - 1) \times (n_x - 1)$ submatrix, A' that has -4 on the main diagonal, and ones on either side of the main diagonal.

$$A' = \begin{bmatrix} -4 & 1 & 0 & \dots & 0 \\ 1 & -4 & 1 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & -4 & 1 \\ 0 & \cdots & 0 & 1 & -4 \end{bmatrix}$$

3. As before, set A' on the main diagonal of A. On either side of the main diagonal, we see the $(n_x - 1) \times (n_x - 1)$ identity matrix, I. The rest of the entries are the $(n_x - 1) \times (n_x - 1)$ zero matrix.

$$A = \begin{bmatrix} A' & I & 0 & \cdots & 0 \\ I & A' & I & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I & A' & I \\ 0 & \cdots & 0 & I & A' \end{bmatrix}$$

2.3.2 Boundary Condition Vector

Note how we can iterate through $\vec{b}$ and $\vec{q}$. We can find i and l for every $v_r = u_{il}$ as:

$$i = (r-1) \pmod{(n_x - 1)} + 1 \qquad l = \left\lfloor \frac{r-1}{n_x - 1} \right\rfloor + 1 \tag{11}$$

We can redefine the pattern for construction $\vec{b}$ in any rectangular domain as:

1. In our rectangular plane, there are $(n_x - 1)(n_y - 1)$ internal points, so $\vec{b}$ has length $(n_x - 1)(n_y - 1)$. We will split this vector into $(n_y - 1)$ subvectors of length $(n_x - 1)$.

2. $f_1(x)$ appears in the first $n_x - 1$ points or in the first subvector. This occurs for every v_r such that $v_r = u_{i1}$ or when $l = \left\lfloor \frac{r-1}{n_x - 1} \right\rfloor + 1 = 1$. These points lie next to the $y = 0$ boundary. $f_2(x)$ appears in the last $n_x - 1$ points or in the last subvector. This occurs for every $v_r = u_{i,(n_y-1)}$ or where $l = \left\lfloor \frac{r-1}{n_x - 1} \right\rfloor + 1 = n_y - 1$. These points lie next to the $y = H$ boundary.

3. $g_1(y)$ appears in the first entry of each subvector. This occurs for any $v_r = u_{1l}$ or where $i = (r-1) \pmod{(n_x - 1)} + 1 = 1$. These points lie against the $x = 0$ boundary. $g_2(y)$ appears in the last entry of each subvector. This occurs for any $v_r = (n_x - 1), 1$ where $i = (r-1) \pmod{(n_x - 1)} + 1 = n_x - 1$. These points lie against the $x = L$ boundary.

2.4 2D Program

Using the patterns found above, a program was created in Matlab to approximate solutions for Poisson's equation. In order to run the program, one would need the following input:

- L, the length of our surface along the x-axis.

- H, the height of our surface along the y-axis.
- $\Delta x = \Delta y$, the size of the subintervals. The number of subintervals, n, will be dictated by L, H, and the size of the subinvervals. In order to have consistently sized subintervals across the region, Δx should evenly divide L and H.
- $Q(x, y)$, the source function across our domain.
- $f_1(x)$, $f_2(x)$, $g_1(y)$, and $g_2(y)$, the boundary conditions of our region.

The program takes these inputs and uses an algorithm based on the patterns above to create A, $\vec{b}$, and $\vec{q}$. It then finds and outputs $\vec{v}$, the approximate solutions at the internal points of the plane. To view the results, one can create a three-dimensional graph with the output.

2.5 Examples

In order to test this program, I compared exact solutions to the program's approximate solutions. With any known three-dimensional function, $u(x, y)$ and a region, one can find the source function and boundary condition functions needed to run the program. Additionally, we must choose Δx, which will dictate how many subintervals our region is divided into. As the number of subintervals n increases, the solution becomes more accurate, but the program takes longer to run. To evaluate how well the program approximates a function, one may compare the approximate solution to a graph of the exact solution $u(x, y)$.

Example 2.1.

$$u(x, y) = x(1 - x)y^2$$

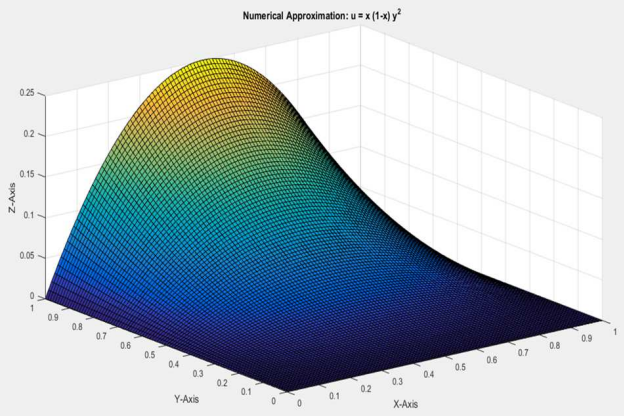
We will approximate this equation in a 1×1 square. This three dimensional equation satisfies

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = Q(x, y) = -2y^2 + 2x(1 - x)$$

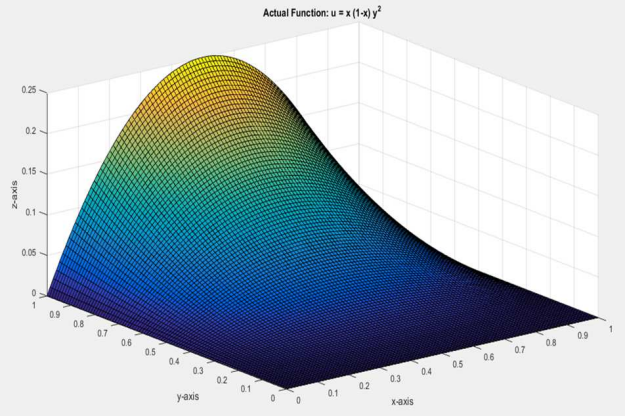
The boundary conditions are given where $x = 0$, $y = 0$, $x = 1$, and $y = 1$.

$$u(x, 0) = f_1(x) = 0 \qquad u(0, y) = g_1(y) = 0$$
$$u(x, 1) = f_2(x) = x(1 - x) \qquad u(1, y) = g_2(y) = 0$$

Let $\Delta x = \Delta y = 0.01$, so there are $n = 100$ subintervals along the x-axis and y-axis. Compare the numerical approximation to the exact solution below.



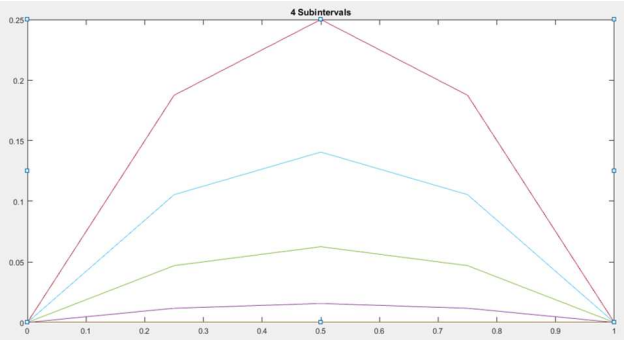
(a) Numerical Approximation



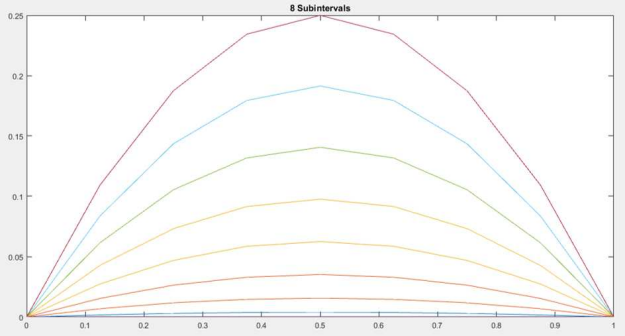
(b) Exact Solution

Figure 3: Exact and Approximate Solution of $u(x, y) = x(1 - x)y^2$

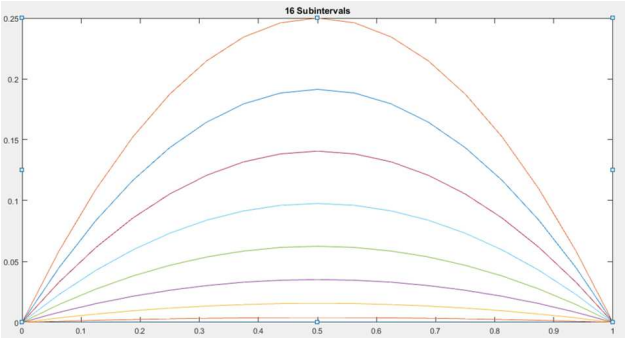
Consider how the number of subintervals affects the accuracy of this approximation. Below, approximate solutions of different subintervals are projected onto the y-axis for a clear view of the parabolas. As the number of subinvtervals increase, and as Δx and Δy decrease, the parabolas get smoother, like the parabolas in the exact solution.



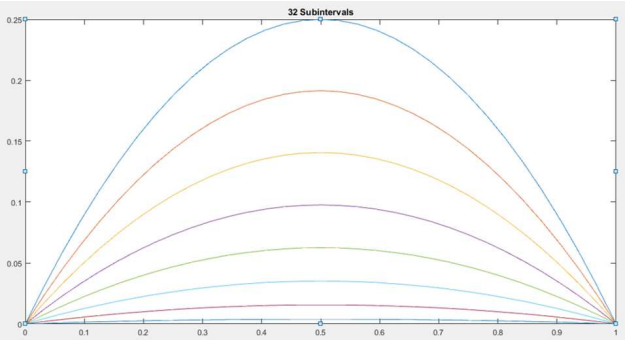
(a) 4 Subintervals



(b) 8 Subintervals



(a) 16 Subintervals



(b) 32 Subintervals

Figure 5: Approximate Solution of $u(x, y) = x(1 - x)y^2$ with Different Subintervals

For small intervals such as $n = 4$ and $n = 8$, the parabolas in the appoximate solutions are very choppy. As n increases, the parabolas in our approximation smooth out, like the parabolas in the actual solution.

Example 2.2. Now let us consider an example in a rectangular domain. Let $L = 3$ and $H = 1$, so that $0 \leq x \leq 3$ and $0 \leq y \leq 1$, and let $\Delta x = \Delta y = 0.05$ so that $n_x = 60$ and $n_y = 20$. We will approximate

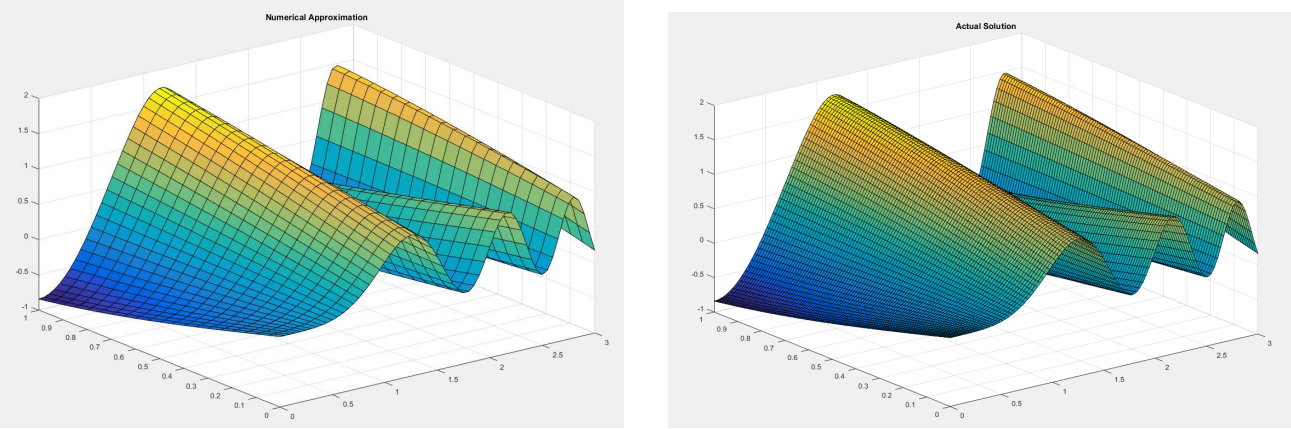
$$u(x, y) = \sin^2(x^2) - \cos(3x) \cdot \sin y$$

We can find the source function and boundary conditions in this domain as:

$$Q(x, y) = 10\cos(3x)\sin y + 8x^2\cos^2(x^2) - 8x^2\sin^2(x^2) + 4\cos(x^2)\sin(x^2)$$

$$u(x, 0) = f_1(x) = \sin^2(x^2) \qquad u(0, y) = g_1(y) = -\sin y$$
$$u(x, 1) = f_2(x) = \sin^2(x^2) - \cos(3x)\sin(H) \qquad u(3, y) = g_2(y) = \sin^2(L^2) - \cos(3L)\sin y$$

Compare the numerical approximation with a graph of the exact solution.



(a) Numerical Approximation

(b) Exact Solution

2.6 Application and Experiment

Poisson's Equation has many applications, such as modeling equilibrium temperature distribution, concentration of particles after diffusion, and potential energy fields, such as electrostatic potential and Newtonian gravity potential [3]. The variation of Poisson's Equation that models heat distribution is

$$\nabla^2 u = -\frac{Q(x, y)}{K_0} \tag{12}$$

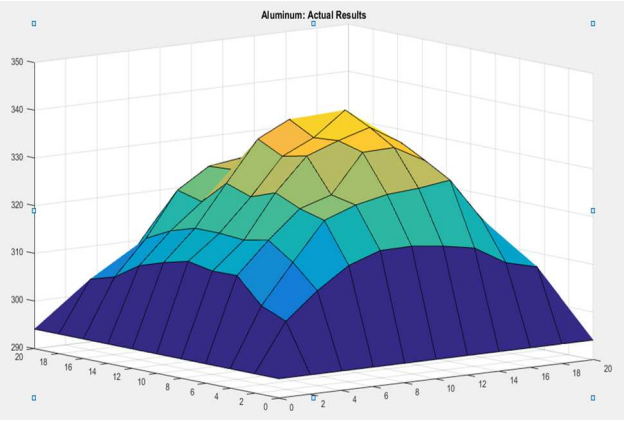
Here, u is a function of temperature across the surface, Q is the heat generated at any point, and K_0 is the thermal conductivity of the material, in other words, how well it conducts heat. K_0 can be a function of x and y if the material is not uniform, but we will consider only uniform material, so the thermal conductivity is constant across the surface.

In order to apply the numerical approximation to a real world example, a 20 by 20 centimeter piece of aluminum foil was heated on a hot plate. The temperature of the room was $294.1K$, which will act as the boundary conditions of our region. The heat of the hot plate was $321.75K$, which

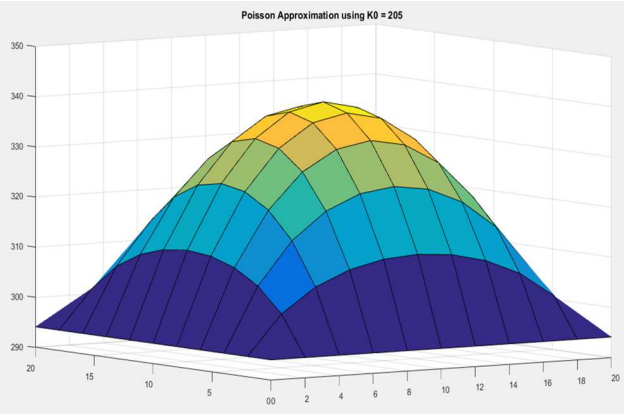
represents $Q(x, y)$, the heat generated at all internal points. The thermal conductivity of aluminum is $205.0K$. Under these conditions, equation (12) becomes

$$\nabla^2 u = -\frac{321.75}{205.0}$$

The boundary conditions are given as $f_1(x) = f_2(x) = g_1(y) = g_2(y) = 294.1K$. We allowed the aluminum foil to sit on the hot plate until it reached an equilibrium temperature. Then, we measured the temperature every centimeter up and across the surface. Below, the figure on the left depicts the temperatures collected in the experiment. On the right, the graph depicts the program's approximate solutions given the boundary conditions and source function.



(a) Temperatures from Lab



(b) Approximate Temperature Distribution

Thus the numerical approximation is equipped to answer real world questions concerning heat flow and temperature. Our approximation shows temperatures that are slightly higher than the temperatures collected in the lab. Note that Poisson's Equation is derived under the assumption that the lateral sides of the surface are perfectly insulated. In the experiment, the aluminum foil released heat into the air.

3 Three-Dimension Case

3.1 Finite Difference Approximations in 3D

We will find solutions to the three-dimensional case using a similar method to the two-dimensional case. Let our domain be a rectangular prism of length L, height H, and width W. Consider how equation (1) changes in three dimensions.

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = Q(x, y, z) \tag{13}$$

1. u and Q are now functions of three variables: $u(x, y, z)$ and $Q(x, y, z)$.
2. We must consider two additional boundary conditions, where $z = 0$ and $z = W$. Our boundary conditions are now denoted as

$$\begin{array}{lll} u(x, 0, z) = f_1(x, z) & u(0, y, z) = g_1(y, z) & u(x, y, 0) = h_1(x, y) \\ u(x, H, z) = f_2(x, z) & u(L, y, z) = g_2(y, z) & u(x, y, W) = h_2(x, y) \end{array}$$

As before, we approximate the second derivatives of u with a finite difference approximation.

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \approx & \frac{u(x_i + \Delta x, y_l, z_k) + u(x_i - \Delta x, y_l, z_k) - 2u(x_i, y_l, z_k)}{(\Delta x)^2} \\ & + \frac{u(x_i, y_l + \Delta y, z_k) + u(x_i, y_l - \Delta y, z_k) - 2u(x_i, y_l, z_k)}{(\Delta y)^2} \\ & + \frac{u(x_i, y_l, z_k + \Delta z) + u(x_i, y_l, z_k - \Delta z) - 2u(x_i, y_l, z_k)}{(\Delta z)^2}\end{aligned} \tag{14}$$

Again, let $\Delta x = \Delta y = \Delta z$. We will apply the finite difference approximation (14) to Poisson's equation (13).

$$\begin{aligned}& u(x_i + \Delta x, y_l, z_k) + u(x_i - \Delta x, y_l, z_k) \\ & \quad + u(x_i, y_l + \Delta y, z_k) + u(x_i, y_l - \Delta y, z_k) \\ & \quad + u(x_i, y_l, z_k + \Delta z) + u(x_i, y_l, z_k - \Delta z) \\ & \quad - 6u(x_i, y_l, z_k) \\ & = (\Delta x)^2 \cdot Q(x, y, z)\end{aligned} \tag{15}$$

Let us apply a similar notation to equation (15) that we used in the two dimensional case. Let $x_i + \Delta x = x_{i+1}$, $y_l + \Delta y = y_{l+1}$, and $z_k + \Delta z = z_{k+1}$. Then, let every $u(x_i, y_l, z_k) = u_{ilk}$. We can simplify equation (15) as:

$$u_{i+1,l,k} + u_{i-1,l,k} + u_{i,l+1,k} + u_{i,l-1,k} + u_{i,l,k+1} + u_{i,l,k-1} - 6u_{i,l,k} = (\Delta x)^2 \cdot Q(x_i, y_l, z_k) \tag{16}$$

3.2 Cube Domains

First, let us consider the case where $H = L = W$. Let $\Delta x = \Delta y = \Delta z$ and let $n = \frac{L}{\Delta x} = \frac{H}{\Delta y} = \frac{W}{\Delta z}$. Consider a sketch of a region where $n = 4$. Here, we can view cross sections of our cube at different z values.

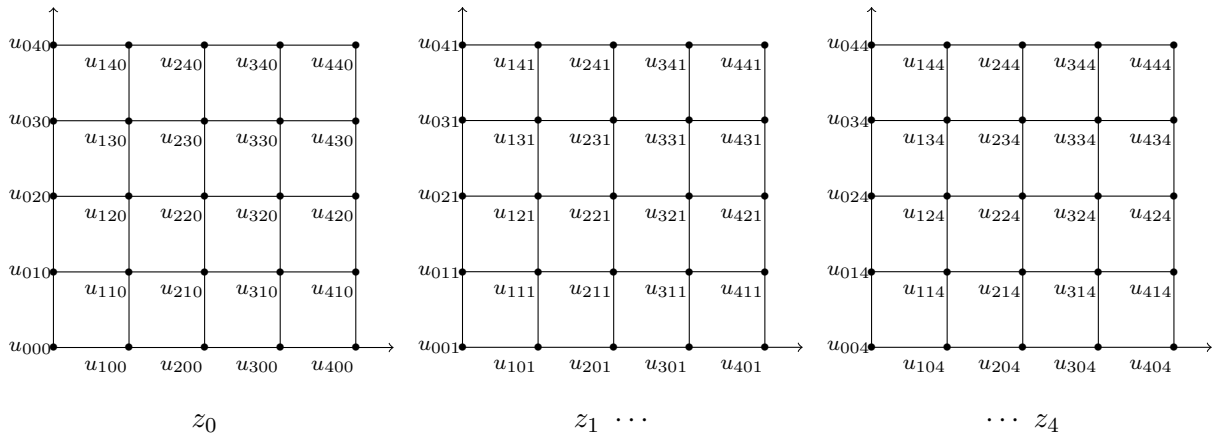


Figure 8: Cube Separated into 4 Subintervals

As before, we can solve for an approximate u_{ilk} by converting equation (16) to a system of linear equations where $0 < i < n$, $0 < l < n$, and $0 < k < n$. Where $n = 4$, we need a system of

$(n-1)^3 = 27$ equations, one for each point inside our cube. For example, we can start these equations as

$$
\begin{array}{lll}
\underline{k=1} & & \\
\underline{l=1} & i=1 & u_{211}+u_{011}+u_{121}+u_{101}+u_{112}+u_{110}-6u_{111}=(\Delta x)^2\cdot Q(x_1,y_1,z_1) \\
 & i=2 & u_{311}+u_{111}+u_{221}+u_{201}+u_{212}+u_{210}-6u_{211}=(\Delta x)^2\cdot Q(x_2,y_1,z_1) \\
 & i=3 & u_{411}+u_{211}+u_{321}+u_{301}+u_{312}+u_{310}-6u_{311}=(\Delta x)^2\cdot Q(x_3,y_1,z_1) \\
\underline{l=2} & i=1 & u_{221}+u_{021}+u_{131}+u_{111}+u_{122}+u_{120}-6u_{121}=(\Delta x)^2\cdot Q(x_1,y_2,z_1) \\
 & i=2 & u_{321}+u_{121}+u_{231}+u_{211}+u_{222}+u_{220}-6u_{221}=(\Delta x)^2\cdot Q(x_2,y_2,z_1) \\
 & i=3 & u_{421}+u_{221}+u_{331}+u_{311}+u_{322}+u_{320}-6u_{321}=(\Delta x)^2\cdot Q(x_3,y_2,z_1) \\
\underline{l=3} & i=1 & u_{231}+u_{031}+u_{141}+u_{121}+u_{132}+u_{130}-6u_{131}=(\Delta x)^2\cdot Q(x_1,y_3,z_1) \\
 & i=2 & u_{331}+u_{131}+u_{241}+u_{221}+u_{232}+u_{230}-6u_{231}=(\Delta x)^2\cdot Q(x_2,y_3,z_1) \\
 & i=3 & u_{431}+u_{231}+u_{341}+u_{321}+u_{332}+u_{330}-6u_{331}=(\Delta x)^2\cdot Q(x_3,y_3,z_1)
\end{array}
$$

$$\vdots$$

$$
\begin{array}{lll}
\underline{k=3} & & \\
\underline{l=3} & i=1 & u_{233}+u_{033}+u_{143}+u_{123}+u_{134}+u_{132}-6u_{133}=(\Delta x)^2\cdot Q(x_1,y_3,z_3) \\
 & i=2 & u_{333}+u_{133}+u_{243}+u_{223}+u_{234}+u_{232}-6u_{233}=(\Delta x)^2\cdot Q(x_2,y_3,z_3) \\
 & i=3 & u_{433}+u_{233}+u_{343}+u_{323}+u_{334}+u_{332}-6u_{333}=(\Delta x)^2\cdot Q(x_3,y_3,z_3)
\end{array}
$$

We will replace u_{ilk} with the necessary boundary conditions. For instance, $u_{0lk} = g_1(y_l,\ z_k)$, $u_{nlk} = g_2(y_l,\ z_k)$, $u_{i0k} = f_1(x_i,\ z_k)$, $u_{ink} = f_2(x_i,\ z_k)$, $u_{il0} = h_1(x_i,\ y_l)$, and $u_{iln} = h_2(x_i,\ y_l)$ are known boundary conditions. All other solutions of internal points will be denoted as v_r. We can turn this system of linear equations into corresponding matrices and vectors as

$$A\vec{v} + \vec{b} = (\Delta x)^2 \cdot \vec{q}$$

Where $\vec{v}$ is the vector of approximate solutions at each point, A is the coefficient matrix of these solutions, $\vec{b}$ is the vector of boundary conditions at these points, and $\vec{q}$ is the source function. Although this equation is the same as the two dimensional case, the A coefficient matrix and boundary condition vector $\vec{b}$ will have different patterns.

3.2.1 A Matrix Pattern

In order to better understand the pattern for the matrix A for all n, consider the matrix A, for small number of subintervals, $n = 4$. It is easiest to evaluate this pattern when you consider A as a block matrix of block matrices.

$$A = \left[\begin{array}{ccc|ccc|ccc||ccc|ccc|ccc||ccc|ccc|ccc}
-6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & -6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 1 & 0 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 0 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -6 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 0 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -6 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -6
\end{array}\right]$$

Consider that A is a block matrix of block matrices. A is a $3 \times 3 = (n-1) \times (n-1)$ matrix of $3 \times 3 = (n-1) \times (n-1)$ block matrices. These block submatrices hold $3 \times 3 = (n-1) \times (n-1)$ submatrices. On the main diagonal of A, we see a notable block matrix A'. On the main diagonal of A' is a matrix A'' with -6 along the main diagonal and 1 on either side. In either side of the main diagonal in A' is the $(n-1) \times (n-1)$ identity matrix, I'.

$$A'' = \begin{bmatrix} -6 & 1 & 0 \\ 1 & -6 & 1 \\ 0 & 1 & -6 \end{bmatrix} \Rightarrow A' = \begin{bmatrix} A'' & I' & 0 \\ I' & A'' & I' \\ 0 & I' & A'' \end{bmatrix}$$

In A, A' is on the main diagonal. On either side of A', is a block matrix I that has $(n-1) \times (n-1)$ identity matrices along main diagonal. In other words, A can be simplified as

$$A = \begin{bmatrix} A' & I & 0 \\ I & A' & I \\ 0 & I & A' \end{bmatrix}$$

The algorithm to form A for any number of subintervals n:

1. A is a $(n-1) \times (n-1)$ block matrix of $(n-1) \times (n-1)$ block matrices. These block submatrices hold $(n-1) \times (n-1)$ submatrices.

2. Define a notable $(n-1) \times (n-1)$ submatrix, A'', with -6 on the main diagonal and 1 on either side.
$$A'' = \begin{bmatrix} -6 & 1 & 0 & \cdots & 0 \\ 1 & -6 & 1 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 1 & -6 & 1 \\ 0 & \cdots & 0 & 1 & -6 \end{bmatrix}$$

3. Now define A' as a $(n-1) \times (n-1)$ block matrix comprised of $(n-1) \times (n-1)$ submatrices. A'' appears on the main diagonal of A'. On either side of A'' is the $(n-1) \times (n-1)$ identity matrix, I'. Here, 0 represents the $(n-1) \times (n-1)$ zero matrix.
$$A' = \begin{bmatrix} A'' & I' & 0 & \cdots & 0 \\ I' & A'' & I' & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I' & A'' & I' \\ 0 & \cdots & 0 & I' & A'' \end{bmatrix}$$

4. Define I to be a $(n-1) \times (n-1)$ block matrix with I' along the main diagonal, and all other entries the $(n-1) \times (n-1)$ zero matrix.
$$I = \begin{bmatrix} I' & 0 & 0 & \cdots & 0 \\ 0 & I' & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I' & 0 \\ 0 & \cdots & 0 & 0 & I' \end{bmatrix}$$

5. Let A be a $(n-1)\times(n-1)$ block matrix with A' across the diagonal. I lies on either side of the main diagonal. Here, 0 represents an $(n-1)\times(n-1)$ block matrix comprised of $(n-1)\times(n-1)$ zero matrices.

$$A = \begin{bmatrix} A' & I & 0 & \cdots & 0 \\ I & A' & I & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I & A' & I \\ 0 & \cdots & 0 & I & A' \end{bmatrix}$$

3.2.2 Boundary Condition Vector Pattern

As before, we have the case that u_{0lk}, u_{nlk}, u_{i0k} , u_{ink}, u_{il0}, and u_{iln} are already defined by the boundary conditions given in the problem. We can separate these boundary conditions from the linear equation to form an $(n-1)^3$ boundary condition vector. Consider that $\vec{b}$ is a block vector of block vectors.

$$\vec{b} = \left(\begin{array}{c} f_1(x_1,z_1)+g_1(y_1,z_1)+h_1(x_1,y_1) \\ f_1(x_2,z_1)+h_1(x_2,y_1) \\ f_1(x_3,z_1)+g_2(y_1,z_1)+h_1(x_3,y_1) \\ \hline g_1(y_2,z_1)+h_1(x_1,y_1) \\ h_1(x_2,y_2) \\ g_2(y_2,z_1)+h_1(x_3,y_2) \\ \hline f_2(x_1,z_1)+g_1(y_3,z_1)+h_1(x_1,y_3) \\ f_2(x_2,z_1)+h_1(x_2,y_3) \\ f_2(x_3,z_1)+g_2(y_3,z_1)+h_1(x_3,u_3) \\ \hline\hline f_1(x_1,z_2)+g_1(y_2,z_2) \\ f_1(x_2,z_2) \\ f_1(x_3,z_2)+g_2(y_1,z_2) \\ \hline g_1(y_2,z_2) \\ 0 \\ g_2(y_2,z_2) \\ \hline f_2(x_1,z_2)+g_1(y_3,z_2) \\ f_2(x_2,z_2) \\ f_2(x_3,z_2)+g_2(y_3,z_2) \\ \hline\hline f_1(x_1,z_3)+g_1(y_1,z_3)+h_2(x_1,y_1) \\ f_1(x_2,z_3)+h_2(x_2,y_1) \\ f_1(x_3,z_3)+g_2(y_1,z_3)+h_2(x_3,y_1) \\ \hline g_1(y_2,z_3)+h_2(x_1,y_2) \\ h_2(x_2,y_2) \\ g_2(y_2,z_3)+h_2(x_3,y_2) \\ \hline f_2(x_1,z_3)+g_1(y_3,z_3)+h_2(x_1,y_3) \\ f_2(x_2,z_3)+h_2(x_2,y_3) \\ f_2(x_3,z_3)+g_2(y_3,z_3)+h_2(x_3,y_3) \end{array}\right)$$

Let us start by observing patterns in $\vec{b}$ where $n=4$. If $\vec{b}$ is an $(n-1)$ vector of block vectors, then $h_1(x,y)$ appears in the first block vector, while $h_2(x,y)$ appears in the last block vector. Within these block vectors, we find a very similar pattern to the two dimensional boundary condition

vector. In each block vector, the first subvector contains all instances of $f_1(x,z)$ while the last subvector contains all instances of $f_2(x,z)$. In each subvector, $g_1(y,z)$ appears in the first entry, while $g_2(y,z)$ appears in the last entry.

For any $v_r = u_{ilk} = u(x_i, y_l, z_k)$, we can find i, l, and k as

$$i = (r-1) \pmod{(n-1)} + 1 \qquad l = \left\lfloor \frac{r-1}{n-1} \right\rfloor \pmod{(n-1)} + 1 \qquad k = \left\lfloor \frac{r-1}{(n-1)^2} \right\rfloor + 1$$

We can form an algorithm to create $\vec{b}$ for any number of subintervals n.

1. $\vec{b}$ is a block vector of length $(n-1)$. Entries in this vector are also block vectors of length $(n-1)$ that contain vectors of length $(n-1)$.

2. $h_1(x_i, y_l)$ appears in the first block vector in $\vec{b}$. This occurs when $v_r = u_{il1}$ or where $k = \left\lfloor \frac{r-1}{(n-1)^2} \right\rfloor + 1 = 1$. Note that inside our cube, these points are directly above the $z = 0$ boundary. $h_2(x_i, y_l)$ appears in the last block vector. This occurs when $v_r = u_{il(n-1)}$, or when $k = \left\lfloor \frac{r-1}{(n-1)^2} \right\rfloor + 1 = n-1$. These points lie at the top of our cube by the boundary $z = W$.

3. Now consider the block vectors within the block vectors in $\vec{b}$. In each block vector, $f_1(x,z)$ appears in the first subvector. This occurs when $v_r = u_{i1k}$ or where $l = \left\lfloor \frac{r-1}{n-1} \right\rfloor \pmod{(n-1)} + 1 = 1$. These points lie directly next to the $y = 0$ boundary. Similarly, $f_2(x,z)$ appears in the last subvector within each block vector. This occurs when $v = u_{i(n-1)k}$, or where $l = \left\lfloor \frac{r-1}{n-1} \right\rfloor \pmod{(n-1)} + 1 = n-1$. These points lie beside the $y = H$ boundary.

4. Within each subvector, $g_1(y,z)$ appears in the first entry. This occurs for all solutions $v_r = u_{1lk}$, when $i = (r-1) \pmod{(n-1)} + 1 = 1$. These points lie directly next to the $x = 0$ boundary. Similarly, $g_2(y,z)$ appears in the last entry of each subvector. This occurs when $v_r = u_{(n-1)lk}$ or when $i = (r-1) \pmod{(n-1)} + 1 = n-1$. In our cube, these points lie directly next to the $x = L$ boundary.

Based on our algorithm, the general boundary condition vector is depicted below.

$$\vec{b} = \begin{pmatrix} f_1(x_1, z_1) + g_1(y_1, z_1) + h_1(x_1, y_1) \\ f_1(x_2, z_1) + h_1(x_2, y_1) \\ \vdots \\ f_1(x_{n-1}, z_1) + g_2(y_1, z_1) + h_1(x_{n-1}, y_1) \\ \hline \vdots \\ \hline g_1(y_l, z_1) + h_1(x_1, y_l) \\ \vdots \\ g_2(y_l, z_1) + h_1(x_{n-1}, y_l) \\ \hline \vdots \\ \hline f_2(x_1, z_1) + g_1(y_{n-1}, z_1) + h_1(x_1, y_{n-1}) \\ f_2(x_2, z_1) + h_1(x_2, y_{n-1}) \\ \vdots \\ f_2(x_{n-1}, z_1) + g_2(y_{n-1}, z_1) + h_1(x_{n-1}, y_{n-1}) \\ \hline\hline \vdots \\ \hline\hline f_1(x_1, z_{n-1}) + g_1(y_1, z_{n-1}) + h_2(x_1, y_1) \\ f_1(x_2, z_{n-1}) + h_2(x_2, y_1) \\ \vdots \\ f_1(x_{n-1}, z_{n-1}) + g_2(y_1, z_{n-1}) + h_2(x_{n-1}, y_1) \\ \hline \vdots \\ \hline g_1(y_l, z_{n-1}) + h_2(x_1, y_l) \\ \vdots \\ g_2(y_l, z_{n-1}) + h_2(x_{n-1}, y_l) \\ \hline \vdots \\ \hline f_2(x_1, z_{n-1}) + g_1(y_{n-1}, z_{n-1}) + h_2(x_1, y_{n-1}) \\ f_2(x_2, z_{n-1}) + h_2(x_2, y_{n-1}) \\ \vdots \\ f_2(x_{n-1}, z_{n-1}) + g_2(y_{n-1}, z_{n-1}) + h_2(x_{n-1}, y_{n-1}) \end{pmatrix}$$

The source vector, $\vec{q}$ does not change from the two dimensional case, if only in that $Q(x_i, y_l, z_k)$ is a function of three variables as opposed to two variable. For example,

$$\vec{q} = \begin{pmatrix} Q(x_1, y_1, z_1) \\ Q(x_2, y_1, z_1) \\ \vdots \\ Q(x_3, y_3, z_3) \end{pmatrix}$$

3.3 Rectangular Prism Domains

Now, let us consider a more general case, where $L = H = W$ is not necessarily true. Let us consider different rectangular prism domains in order to identify patterns in the A matrix and boundary condition vector. We will keep $\Delta x = \Delta y = \Delta z$. We can define the number of subintervals along

different axes as $n_x = \frac{L}{\Delta x}$, $n_y = \frac{H}{\Delta y}$, and $n_z = \frac{W}{\Delta z}$. Let us consider two cases, alternating the values for n_x, n_y, and n_z.

3.3.1 A Matrix Pattern

1. Consider a rectangular prism with length $L = 3$, height $H = 4$ and width $W = 5$. Let $\Delta x = \Delta y = \Delta z = 1$, so $n_x = \frac{L}{\Delta x} = 3$, or that the length of our rectangular prism will be separated into 3 subintervals. Similarly, $n_y = 4$ and $n_z = 5$. Here we have $(n_x - 1)(n_y - 1)(n_z - 1) = (3 - 1)(4 - 1)(5 - 1) = 2 \cdot 3 \cdot 4 = 24$ internal points of our domain.

2. Now, consider a similar case, where $L = 5$, $H = 3$, $W = 4$, and $\Delta x = \Delta y = \Delta z = 1$. Now, $n_x = 5$, $n_y = 3$, and $n_z = 4$.

The matrices for these two cases are depicted in the following pages. We will note how the pattern of the A matrix and $\vec{b}$ vector change with different subintervals along the x, y, and z axes.

Case 1: $n_x = 3$, $n_y = 4$, and $n_z = 5$

$$A_1 = \left[\begin{array}{cc|cc|cc||cc|cc|cc||cc|cc|cc||cc|cc|cc}
-6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 1 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline\hline
1 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 1 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -6 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & -6
\end{array}\right]$$

Note that A_1 is a $4 \times 4 = (n_z - 1) \times (n_z - 1)$ block matrix of $3 \times 3 = (n_y - 1) \times (n_y - 1)$ matrices with $2 \times 2 = (n_x - 1) \times (n_x - 1)$ submatrices as entries. Using a similar definition for A' and I as in the cube domain cases, we find that A_1 can be simplified as

$$A_1 = \begin{bmatrix} A' & I & 0 & 0 \\ I & A' & I & 0 \\ 0 & I & A' & I \\ 0 & 0 & I & A' \end{bmatrix}$$

Case 2: $n_x = 5$, $n_y = 3$, and $n_z = 4$.

$$A_2 = \left[\begin{array}{cccc|cccc||cccc|cccc||cccc|cccc||cccc}
-6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -6 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & 0 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -6 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -6 & 0 & 0 & 0 & 1 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -6 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -6
\end{array}\right]$$

Note that A_2 is a $3 \times 3 = (n_z - 1) \times (n_z - 1)$ block matrix of $2 \times 2 = (n_y - 1) \times (n_y - 1)$ matrices with $4 \times 4 = (n_x - 1) \times (n_x - 1)$ submatrices as entries. A_2 can be simplified as:

$$A_2 = \begin{bmatrix} A' & I & 0 \\ I & A' & I \\ 0 & I & A' \end{bmatrix}$$

Let us identify the pattern as n_x, n_y, and n_z change to form a general algorithm for constructing A.

1. A is a $(n_z-1)\times(n_z-1)$ block matrix of $(n_y-1)\times(n_y-1)$ matrices that have entries of $(n_x-1)\times(n_x-1)$ matrices.

2. Define a notable $(n_x-1)\times(n_x-1)$ submatrix, A'', with -6 on the main diagonal and 1 on either side.
$$A' = \begin{bmatrix} -6 & 1 & 0 & \cdots & 0 \\ 1 & -6 & 1 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 1 & -6 & 1 \\ 0 & \cdots & 0 & 1 & -6 \end{bmatrix}$$

3. Define A' as a $(n_y-1)\times(n_y-1)$ submatrix comprised of $(n_x-1)\times(n_x-1)$ submatrices. A'' appears on the main diagonal of A'. On either side of A'' is the $(n_x-1)\times(n_x-1)$ identity matrix, I'. Here, 0 represents the $(n_x-1)\times(n_x-1)$ zero matrix.
$$A' = \begin{bmatrix} A'' & I' & 0 & \cdots & 0 \\ I' & A'' & I' & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I' & A'' & I' \\ 0 & \cdots & 0 & I' & A'' \end{bmatrix}$$

4. Define I to be a $(n_y-1)\times(n_y-1)$ block matrix with I' along the main diagonal, and all other entries the $(n_x-1)\times(n_x-1)$ zero matrix.
$$I = \begin{bmatrix} I' & 0 & 0 & \cdots & 0 \\ 0 & I' & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I' & 0 \\ 0 & \cdots & 0 & 0 & I' \end{bmatrix}$$

5. Let A be a $(n_z-1)\times(n_z-1)$ block matrix with A' across the diagonal. I lies on either side of the main diagonal. Here, 0 represents an $(n_y-1)\times(n_y-1)$ block matrix of $(n_x-1)\times(n_x-1)$ zero matrices.
$$A = \begin{bmatrix} A' & I & 0 & \cdots & 0 \\ I & A' & I' & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & I & A' & I \\ 0 & \cdots & 0 & I & A' \end{bmatrix}$$

3.3.2 Boundary Condition Vector Pattern

First, we must consider how we iterate through a rectangular prism. For every $v_r = u_{ilk} = u(x_i, y_l, z_k)$, we can find i, l, and k as:

$$i = (r-1) \pmod{(n_x-1)} + 1 \qquad l = \left\lfloor \frac{r-1}{n_x-1} \right\rfloor \pmod{(n_y-1)} + 1 \qquad k = \left\lfloor \frac{r-1}{(n_y-1)(n_x-1)} \right\rfloor + 1$$

Given the two cases above, consider the corresponding boundary condition vectors:

$$b_1 = \begin{pmatrix} f_1(x_1,z_1)+g_1(y_1,z_1)+h_1(x_1,y_1) \\ f_1(x_2,z_1)+g_2(y_1,z_1)+h_1(x_2,y_1) \\ \hline g_1(y_2,z_1)+h_1(x_1,y_2) \\ g_2(y_2,z_1)+h_1(x_2,y_2) \\ \hline f_2(x_1,z_1)+g_1(y_3,z_1)+h_1(x_1,y_3) \\ f_2(x_2,z_1)+g_2(y_3,z_1)+h_1(x_2,y_3) \\ \hline\hline f_1(x_1,z_2)+g_1(y_1,z_2) \\ f_1(x_2,z_2)+g_2(y_1,z_2) \\ \hline g_1(y_2,z_2) \\ g_2(y_2,z_2) \\ \hline f_2(x_1,z_2)+g_1(y_3,z_2) \\ f_2(x_2,z_2)+g_2(y_3,z_2) \\ \hline\hline f_1(x_1,z_3)+g_1(y_1,z_3) \\ f_1(x_2,z_3)+g_2(y_1,z_3) \\ \hline g_1(y_2,z_3) \\ g_2(y_2,z_3) \\ \hline f_2(x_1,z_3)+g_1(y_3,z_3) \\ f_2(x_2,z_3)+g_2(y_3,z_3) \\ \hline\hline f_1(x_1,z_4)+g_1(y_1,z_4)+h_2(x_1,y_1) \\ f_1(x_2,z_4)+g_2(y_1,z_4)+h_2(x_2,y_1) \\ \hline g_1(y_2,z_4)+h_2(x_1,y_2) \\ g_2(y_2,z_4)+h_2(x_2,y_2) \\ \hline f_2(x_1,z_4)+g_1(y_3,z_4)+h_2(x_1,y_3) \\ f_2(x_2,z_4)+g_2(y_3,z_4)+h_2(x_2,y_3) \end{pmatrix}$$

$$b_2 = \begin{pmatrix} f_1(x_1,z_1)+g_1(y_1,z_1)+h_1(x_1,y_1) \\ f_1(x_2,z_1)+h_1(x_2,y_1) \\ f_1(x_3,z_1)+h_1(x_3,y_1) \\ f_1(x_4,z_1)+g_2(y_1,z_1)+h_1(x_4,y_1) \\ \hline f_2(x_1,z_1)+g_1(y_2,z_1)+h_1(x_1,y_2) \\ f_2(x_2,z_1)+h_1(x_2,y_2) \\ f_2(x_3,z_1)+h_1(x_3,y_2) \\ f_2(x_4,z_1)+g_2(y_2,z_1)+h_1(x_4,y_2) \\ \hline\hline f_1(x_1,z_2)+g_1(y_1,z_2) \\ f_1(x_2,z_2) \\ f_1(x_3,z_2) \\ f_1(x_4,z_2)+g_2(y_1,z_2) \\ \hline f_2(x_1,z_2)+g_1(y_2,z_2) \\ f_2(x_2,z_2) \\ f_2(x_3,z_2) \\ f_2(x_4,z_2)+g_2(y_2,z_2) \\ \hline\hline f_1(x_1,z_3)+g_1(y_1,z_3)+h_2(x_1,y_1) \\ f_1(x_2,z_3)+h_2(x_2,y_1) \\ f_1(x_3,z_3)+h_2(x_3,y_1) \\ f_1(x_4,z_3)+g_2(y_1,z_3)+h_2(x_4,y_1) \\ \hline f_2(x_1,z_3)+g_1(y_2,z_3)+h_2(x_1,y_2) \\ f_2(x_2,z_3)+h_2(x_2,y_2) \\ f_2(x_3,z_3)+h_2(x_3,y_2) \\ f_2(x_4,z_3)+g_2(y_2,z_3)+h_2(x_4,y_2) \end{pmatrix}$$

Let us compare the pattern in $\vec{b}_1$ and $\vec{b}_2$. Note that $\vec{b}_1$ is a block vector of length $4 = (n_z - 1)$ that contains block vectors of length $3 = (n_y - 1)$. This block subvector holds subvectors of length $2 = (n_x - 1)$. On the other hand $\vec{b}_2$ is a block vector of length $3 = (n_z - 1)$ that contains block subvectors of length $2 = (n_y - 1)$. These block subvectors hold subvectors of length $4 = (n_x - 1)$. Within both vectors, $\vec{b}_1$ and $\vec{b}_2$, we see the pattern from our cube domain examples. $h_1(x, y)$ appears in the first block vector, or the first $(n_x - 1)(n_y - 1)$ points, while $h_2(x, y)$ appears in the last subvector, or the last $(n_x - 1)(n_y - 1)$ points. In each of these block vectors, $f_1(x, z)$ appears in the first subvector, while $f_2(x, z)$ appears in the last subvector. $g_1(y, z)$ appears in the first entry of each subvector, while $g_2(y, z)$ appears in the last entry.

The algorithm for constructing $\vec{b}$ for any rectangular prism domain:

1. The boundary condition vector $\vec{b}$ has length $(n_x - 1)(n_y - 1)(n_z - 1)$. This vector can be broken into $(n_z - 1)$ block vectors of length $(n_y - 1)$ that hold subvectors of length $(n_x - 1)$.

2. $h_1(x_i, y_l)$ appears in the first block vector in $\vec{b}$. This occurs when $v_r = u_{il1}$ or where $k = \left\lfloor \frac{r-1}{(n_y-1)(n_x-1)} \right\rfloor + 1 = 1$. Note that inside our cube, these points are directly above the $z = 0$ boundary. $h_2(x_i, y_l)$ appears in the last block vector. This occurs when $v_r = u_{il(n_z-1)}$, or when $k = \left\lfloor \frac{r-1}{(n_y-1)(n_x-1)} \right\rfloor + 1 = n_z - 1$. These points lie at the edge of our cube where $z = W$.

3. Now consider the subvectors inside each block vector in $\vec{b}$. In each block vector, $f_1(x,z)$ appears in the first subvector. This occurs when $v_r = u_{i1k}$ or where $l = \left\lfloor \frac{r-1}{n_x-1} \right\rfloor \pmod{(n_y - 1)} + 1 = 1$. These points lie directly next to the boundary condition where $y = 0$. Similarly, $f_2(x,z)$ appears in the last subvector within each block vector. This occurs when $v_r = u_{i(n_y-1)k}$, or where $l = \left\lfloor \frac{r-1}{n_x-1} \right\rfloor \pmod{(n_y - 1)} + 1 = n_y - 1$. These points lie beside the $y = H$ boundary.

4. Similarly, consider the subvectors inside the block vectors of $\vec{b}$. In each subvector, $g_1(y,z)$ appears in the first entry. This occurs for all solutions $v_r = u_{1lk}$ or when $i = (r-1) \pmod{(n_x - 1)} + 1 = 1$. These points lie directly next to the $x = 0$ boundary. Similarly, $g_2(y,z)$ appears in the last entry of each subvector. This occurs when $v_r = u_{(n_x-1)lk}$ or when $i = (r-1) \pmod{(n_x - 1)} + 1 = n_x - 1$. In our cube, these points lie directly next to the $x = L$ boundary.

3.4 3D Program

Using the patterns found above, a program in Matlab was created to approximate the solutions for Poisson's Equation in three dimensions. In order to run the program, one would need the following input:

- L, the length of our region along the x-axis.
- H, the height of our region along the y-axis.
- W, the width of our region along the z-axis.
- $\Delta x = \Delta y = \Delta z$, the size of the subintervals. This will dictate the number of subintervals along each axis.
- $Q(x,y,z)$, source function through our region.
- $f_1(x,z)$, $f_2(x,z)$, $g_1(y,z)$, $g_2(y,z)$, $h_1(x,y)$, and $h_2(x,y)$, the boundary conditions of our region.

The program uses the algorithms defined above to construct A, $\vec{b}$, and $\vec{q}$. Then, it solves for the approximate solution vector $\vec{v}$. To test the results of the program, one can compare the graph of the approximate solution with the exact solution at different z values.

3.5 Examples

Example 3.1. Here, we will evaluate $u(x,y,z)$ in a cube, where $H = L = W = 4$. Let $\Delta x = \Delta y = \Delta x = 0.25$, so that the x, y, and z-axes are separated into $n = 16$ subintervals.

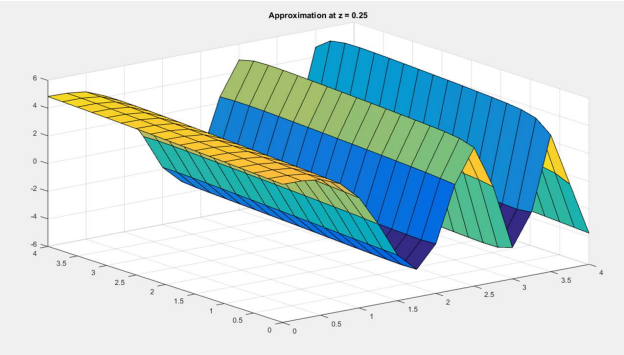
$$u(x,y,z) = 4\cos(x^2 + z^2) + \sin(yz)$$

We can find the source function $Q(x,y,z)$ and the six boundary conditions as:

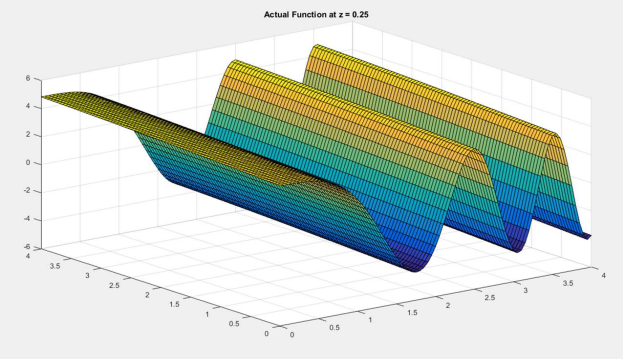
$$Q(x,y,z) = -16\sin(x^2 + z^2) - 16x^2\cos(x^2 + z^2) - 16z^2\cos(x^2 + z^2) - y^2\sin(yz) - z^2\sin(yz)$$

$$f_1(x,z) = 4\cos(x^2+z^2) \qquad f_2(x,z) = 4\cos(x^2+z^2)+\sin(4z)$$
$$g_1(y,z) = 4\cos(z^2)+\sin(yz) \qquad g_2(y,z) = 4\cos(16+z^2)+\sin(yz)$$
$$h_1(x,y) = 4\cos(x^2) \qquad h_2(x,y) = 4\cos(x^2+16)+\sin(16y)$$

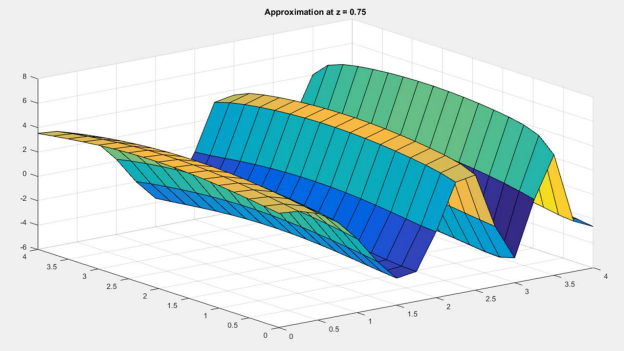
To test the accuracy of the results, consider a few comparisons between the approximate and exact solutions at different values of z.



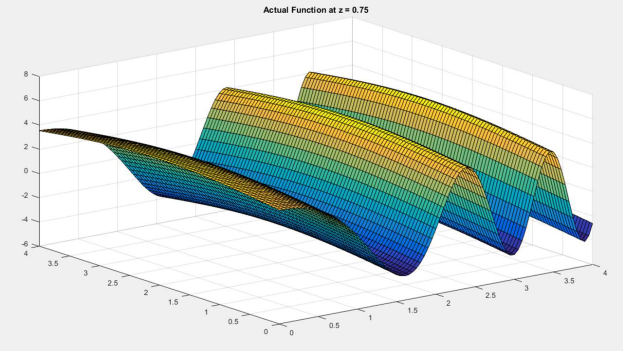
(a) Approximation at $z = 0.25$



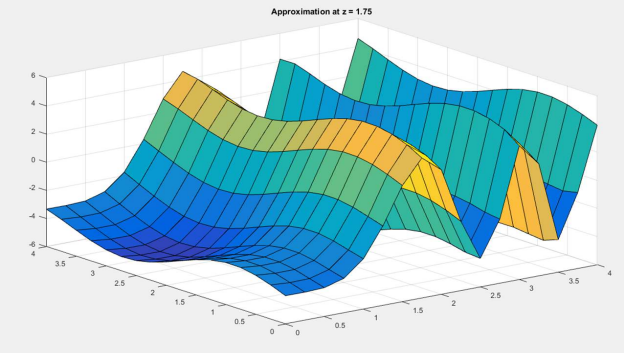
(b) Actual Solution at $z = 0.25$



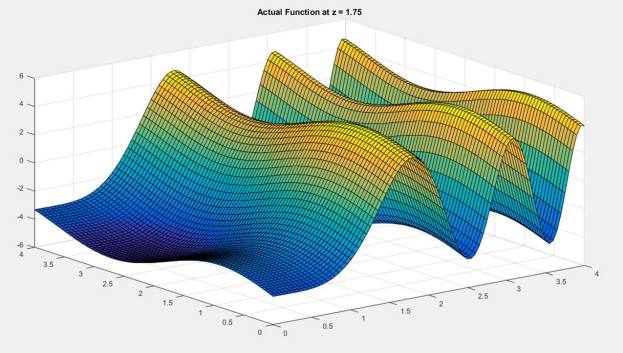
(a) Approximation at $z = 0.75$



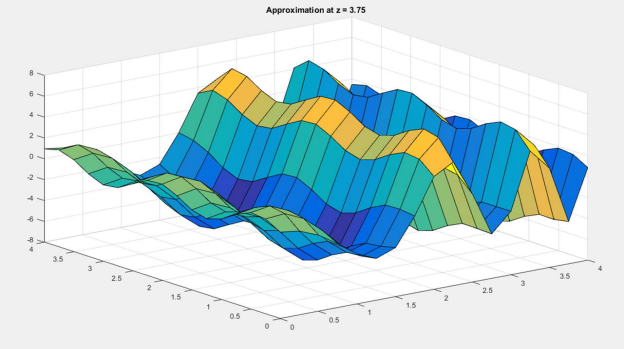
(b) Actual Solution at $z = 0.75$



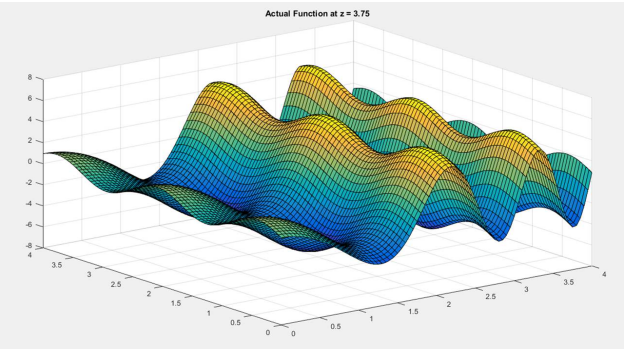
(a) Approximation at $z = 1.75$



(b) Actual Solution at $z = 1.75$



(a) Approximation at $z = 3.75$



(b) Actual Solution at $z = 3.75$

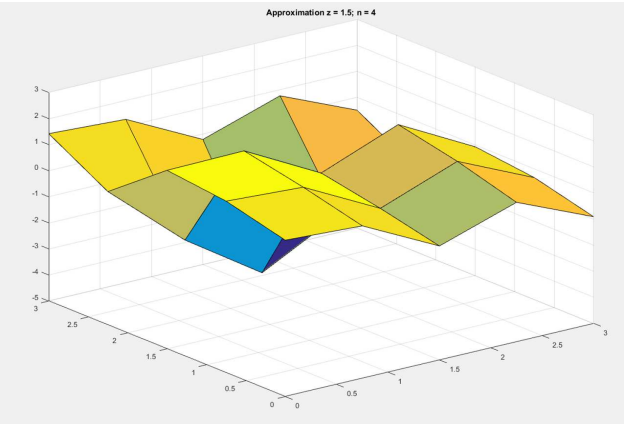
Example 3.2. To show that these approximate solutions get closer to the actual solutions as the number of subintervals increase, consider approximations at increasing subintervals $n = 4,\ 8,\ 16,\ 24$ at a stationary $z = 1.5$ in the following example where $H = L = W = 3$.

$$u(x, y, z) = \cos(x^2) + \sin(y)\cos(z) + \sin(y^2)$$

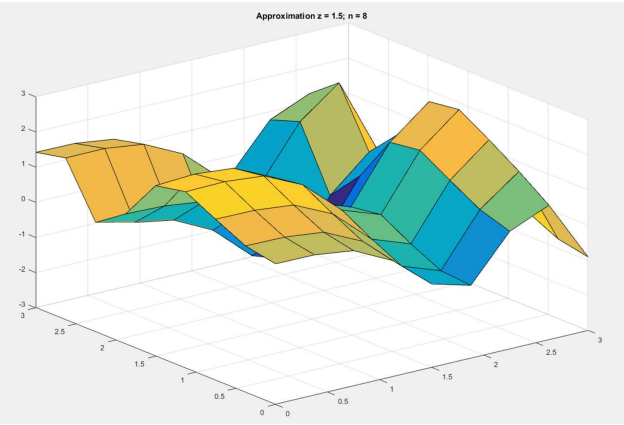
We can find the source function Q and the boundary conditions as:

$$Q(x, y, z) = 2\cos(y^2) - 2\sin(x^2) - 2\cos(z)\sin(y) - 4x^2\cos(x^2) - 4y^2\sin(y^2)$$

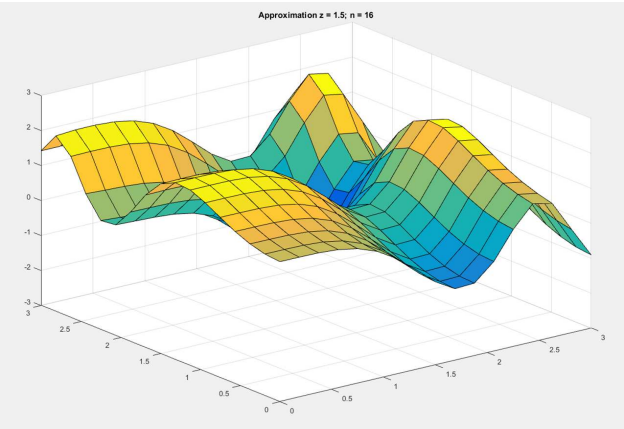
$$\begin{aligned} f_1(x, z) &= \cos(x^2) & f_2(x, z) &= \cos(x^2) + \sin(3)\cos(z) + \sin(9) \\ g_1(y, z) &= 1 + \sin(y)\cos(z) + \sin(y^2) & g_2(y, z) &= \cos(9) + \sin(y)\cos(z) + sin(y^2) \\ h_1(x, y) &= \cos(x^2) + \sin(y) + \sin(y^2) & h_2(x, y) &= \cos(x^2) + \sin(y)\cos(3) + \sin(y^2) \end{aligned}$$



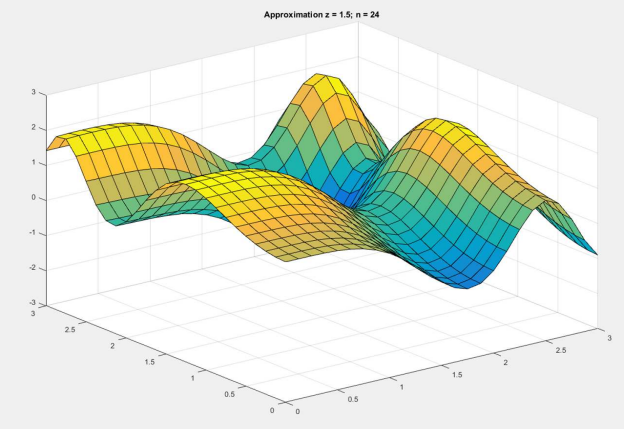
(a) 4 Subintervals



(b) 8 Subintervals



(a) 16 Subintervals



(b) 24 Subintervals

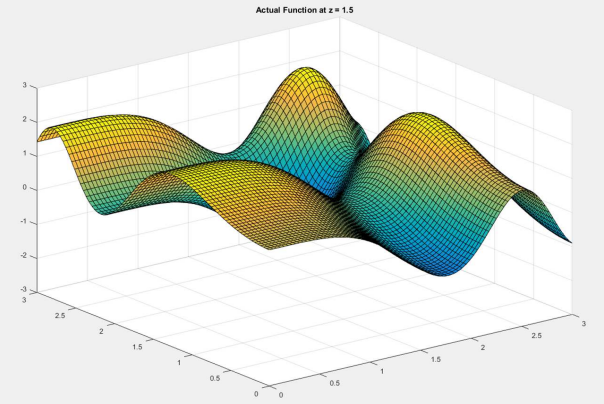


Figure 15: Actual Solution

As the number of subintervals increase, the approximate solution gets closer to the exact solution. When $n = 4$, we see a very inaccurate approximation. If the graph has many direction changes, approximating with a small number of subintervals can be inaccurate and unpredictable. Where $n = 4$, the graph has a drastic, unexpected minimum. As n increases, the curves of the graph become more defined in smooth, like the exact solution.

Example 3.3. Next, let us test results in rectangular prism domains, where $L = 2$, $H = 5$, and $W = 3.5$. We can choose $\Delta x = \Delta y = \Delta z = 0.25$, so that $n_x = 8$, $n_y = 20$, and $n_z = 14$.

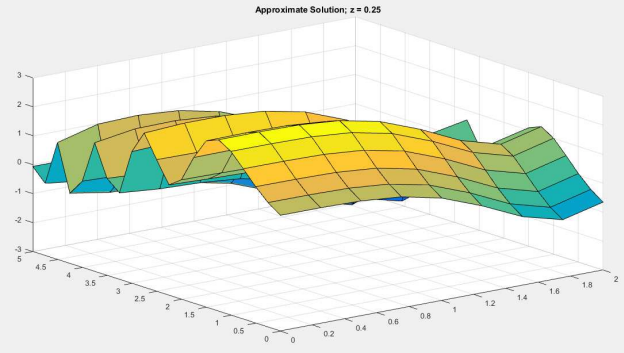
$$u(x, y, z) = \cos(x^2) + \sin(y) \cdot \cos(z) + \sin(y^2)$$

We can find the source function and boundary conditions for $u(x, y, z)$ as

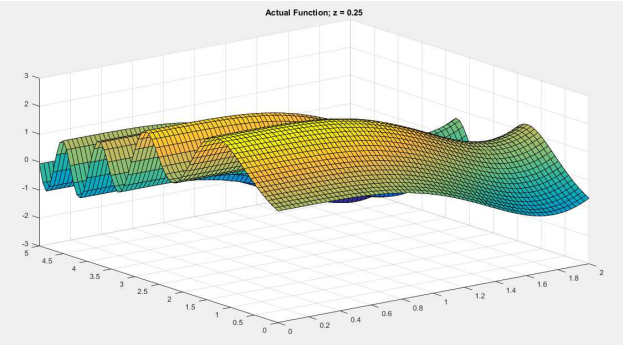
$$Q(x, y, z) = 2\cos(y^2) - 2\sin(x^2) - 2\cos(z)\sin(y) - 4x^2\cos(x^2) - 4y^2\sin(y^2)$$

$$f_1(x, z) = \cos(x^2) \qquad f_2(x, z) = \cos(x^2) + \sin 5 \cdot \cos(z) + \sin(25)$$
$$g_1(y, z) = 1 + \sin(y) \cdot \cos(z) + sin(y^2) \qquad g_2(y, z) = \cos(4) + \sin(y) \cdot \cos(z) + \sin(y^2)$$
$$h_1(x, y) = \cos(x^2) + \sin(y) + sin(y^2) \qquad h_2(x, y) = \cos(x^2) + \sin(y) \cdot \cos(3.5) + \sin(y^2)$$

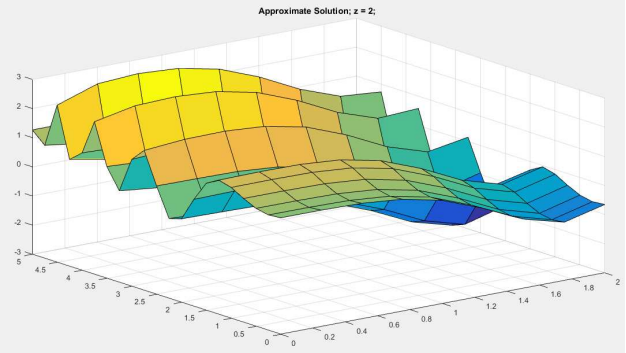
Now, we can compare the numerical approximation with the exact solution at different values of z.



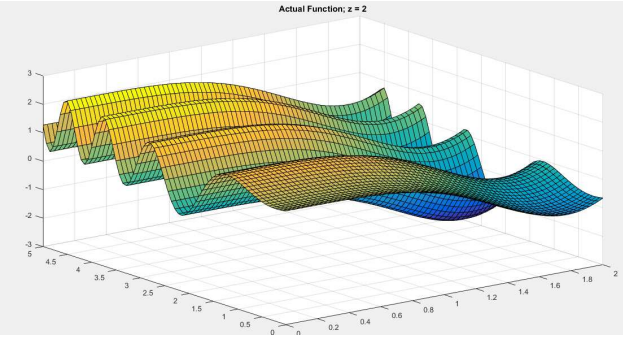
(a) Approximation at $z = 0.25$



(b) Actual Solution at $z = 0.25$



(a) Approximation at $z = 2$



(b) Actual Solution at $z = 2$

4 Conclusion

Although actual solutions to Poisson's equation are difficult to find, we have shown that numerical methods can be used to find accurate approximations. We can approximate Poisson's equation with a finite difference approximation and create a system of equations to solve for solutions at each internal points of our region. By finding solutions by hand, we identified patterns to construct the matrices and vectors needed to approximate solutions in two and three dimensions. These algorithms were used to create programs to find approximations for any number of subintervals. Through many examples, we have shown that the program can accurately approximate 3 and 4 dimensional functions as well as predict equilibrium temperature distribution collected in a lab.

References

[1] Kathryn E. Dillinger, *Analysis of the heat equation with a heat source term*, (2013).

[2] Richard Haberman, *Applied partial differential equations: With fourier series and boundary value problems*, 4 ed., Upper Saddle River, NJ: Pearson Prentice Hall, 2004.

[3] Robert E. Hunt, *Chapter 2: Poisson's equation*, (2007).

[4] Erin L. Strange, *Computation models of the diffusion equation*, (2011).

[5] Teresa G. Yao, *The diffusion of a chemical pollutant modeled by a fourier series*, (2012).